

SRI International

Final Report – 15 September 2004

Mixed-Initiative Planning and Scheduling for Science Missions

SRI Project No.: P11397

Grant No.: NCC 2-1267

Period of Performance: 1 May 2001 through 30 April 2004

Prepared by:

Dr. Karen L. Myers, (PI)
Dr. Michael J. Wolverton
Artificial Intelligence Center
Information and Computing Sciences Division

Approved by:

Dr. C. Raymond Perrault, Director
Artificial Intelligence Center
Information and Computing Sciences Division

Dr. William S. Mark, Vice President
Information and Computing Sciences Division

Prepared for:

Dr. John L. Bresina
NASA/Ames Research Center
Computational Sciences Division
M/S 269-2
Moffett Field, CA 94035-100

Table of Contents

<u>1</u>	<u>Introduction</u>	1
1.1	<u>Background</u>	1
1.2	<u>Technical Approach</u>	1
1.3	<u>Summary of Accomplishments</u>	2
<u>2</u>	<u>Explaining Schedules Using Examples</u>	3
2.1	<u>Overview</u>	4
2.2	<u>Generating Candidate Examples</u>	5
2.3	<u>Selecting Diverse Examples</u>	5
2.4	<u>Justifying Examples</u>	5
2.4.1	<u>Concise Justification</u>	6
2.4.2	<u>Verbose Justification</u>	7
2.5	<u>Sample Explanation</u>	7
2.6	<u>Discussion</u>	8
<u>3</u>	<u>Qualitative Plan Summarization and Comparison</u>	10
3.1	<u>Overview</u>	11
3.2	<u>Technical Background</u>	11
3.3	<u>Domain Metatheory</u>	12
3.3.1	<u>Template Features</u>	13
3.3.2	<u>Task Features</u>	13
3.3.3	<u>Roles</u>	14
3.4	<u>Experimental Framework</u>	14
3.5	<u>Plan Summarization</u>	16
3.5.1	<u>Sample Plan Summary</u>	18
3.6	<u>Comparing Two Plans</u>	22
3.6.1	<u>Feature Differencing</u>	22
3.6.2	<u>Role Differencing</u>	23
3.6.3	<u>Sample Plan Comparison</u>	25
3.7	<u>Plan Space Analysis</u>	27
3.7.1	<u>Identifying Unique Characteristics of a Plan</u>	27
3.7.2	<u>Maximally Different Plans</u>	28
3.8	<u>Discussion</u>	32
<u>4</u>	<u>Conclusions</u>	34
<u>5</u>	<u>Acknowledgments</u>	34
<u>6</u>	<u>References</u>	35

1 Introduction

This document constitutes a final technical report for NASA Contract NCC 2-1267, *Mixed-initiative Planning and Scheduling for Science Missions*.

1.1 Background

The objective of this joint NASA Ames/JPL/SRI project was to develop mixed-initiative planning and scheduling technology that would enable more effective and efficient planning of science missions. The original intent behind the project was to have all three organizations work closely on the overall research and technology development objectives. Shortly after the project began, however, the Ames and JPL project members made a commitment to develop and field an operational mixed-initiative planning and scheduling tool called MAPGEN for the 2003 Mars Exploration Rover (MER) mission [Ai-Chang et al. 2003]. Because of the tremendous amounts of time and effort that went into making that tool a success, the Ames and JPL personnel were mostly unavailable for collaboration on the joint objectives of the original proposal.

Until November of 2002, SRI postponed work on the project in the hope that the Ames and JPL personnel would be able to find time for the planned collaborative research. During discussions between Dr. Karen Myers (the SRI institutional PI) and Dr. John Bresina (the project PI) during November of 2002, it was mutually agreed that SRI should work independently to achieve some of the research objectives for the project. In particular, Dr. Bresina identified explanation of plans and planner behavior as a critical area for research, based on feedback from demonstrating an initial prototype of MAPGEN to the operational community. For that reason, our focus from November of 2002 through the end of the project was on designing explanation methods to address this need.

1.2 Technical Approach

Our work on this project covers two complementary types of explanatory capability, both of which are based on concisely presenting salient portions of the plan space and comparing and contrasting elements of that space. *Element-level explanation* focuses on justifying the system's plan, in specific parts and overall, based on its advantages to plausible alternative solutions. Section 2 summarizes our work in this area. *Plan-level explanation* focuses on summarizations of individual plans and comparisons across plans. Section 3 summarizes our work on plan-level explanation, covering techniques for summarizing individual plans, comparing pairs of plans, and analyzing a space of candidate solution plans.

Most previous work on plan explanation has been grounded in methods that are tightly linked to the syntactic characteristics of a plan's structure or the underlying reasoning processes used to generate it. A key problem with such approaches is that a plan's structure and the reasoning processes used to create it do not generally match the user's conceptualization of the domain. As such, the resultant explanations are of limited value. One common theme in the two lines of work that we performed on this project is the focus on *user-centric* methods that are geared toward improved understandability by an

individual user. This theme is in line with a model of *reconstructive explanation* (Wick and Thompson 1992), in which an explanation is produced not by the system's own internal knowledge, but by a separate store of explanation knowledge designed specifically with the user in mind.

1.3 Summary of Accomplishments

In terms of element-level explanation, we developed a method of explaining schedules based on examples. This approach contrasts with most traditional approaches to explanation, which present the chain of reasoning the system used to produce the solution. In the example-based approach, the explainer justifies the system's reasoning by generating examples of alternative solutions, selecting from these examples to create a diverse set for presentation, and justifying the system's solutions with respect to each of the examples. The approach is implemented in a prototype explanation module called EBE, which generates machine-readable or textual explanations of the outputs of a constraint reasoner.

In terms of plan-level explanation, we defined a domain-independent framework for summarization and comparison that is grounded in the use of a *domain metatheory*. The domain metatheory is an abstract characterization of a planning domain that specifies important semantic differences among operators, planning variables, and instances. This abstraction provides the means to describe plans and their components in high-level, semantically meaningful terms. Based on this framework, we developed a suite of tools that provide three key explanatory capabilities: (a) summarization of an individual plan, (b) comparison of pairs of plans, and (c) analysis of a collection of plans. Finally, we showed the potential for these methods to increase a user's understanding of plans by evaluating the tools within an extensive domain for special operations planning. Section 2.5 describes our work on this topic in further detail.

The intent behind the original proposal was to develop tools that would be integrated into the evolving MAPGEN system. However, SRI was unable to secure a copy of MAPGEN for use in the implementation and evaluation aspects of our work because of factors beyond our control (in particular, problems with releasing the code to an organization outside of NASA). For this reason, we were forced to explore implementations of our ideas on explanation within planning and scheduling systems that had been developed previously at SRI (specifically, SRI's Calendar Manager [Berry et al. 2003] and PASSAT [Myers et al. 2002] systems). The core ideas, however, were developed with an eye toward eventual integration with a system like MAPGEN, namely a combined planning and scheduling system for the development of complex, hierarchical plans.

2 Explaining Schedules Using Examples

In designing an approach to explain schedules, we confronted two major issues. The first is that scheduling presents unique challenges for explanation generation because of the complexity of the reasoning involved. The process of producing a schedule in real-world domains can be extremely complex, involving a very large number of interactions between constraints. While many traditional approaches to AI explanation involve reiterating the system's chain of reasoning in some form [Buchanan & Shortliffe 1984, McGuinness & da Silva 2003] or expanding on that chain of reasoning [Clancey 1983, Swartout et al. 1991], the complexity of constraint reasoning makes the reiteration approach impractical for explaining realistic schedules. The reasoning to be explained becomes even more complex when soft constraints are allowed—that is, when a subset of the constraints are relaxable, and possibly partially ordered by a preference function.

The second issue is the purpose for which the explanations are to be used. There are many possible purposes for explanation in AI systems. These include *teaching* the user to be able to perform the system's task, *proving* to the user that the system's conclusion is correct, *describing* a concept or object used in the system, and establishing *trust* in the user that the system has solved its task correctly or competently. The type of explanation required varies greatly depending on which of these purposes applies.

Our goal is to produce explanations for the last of these purposes. As with the *satisficing* explanations of [Wolverton 1995], the aim of these explanations is to enable the user to trust the system's method and results,¹ rather than exhaustively proving the correctness of the system's conclusion. Fortunately, to establish trust, it is not necessary to reiterate all, or even significant portions, of the system's reasoning. Often a person will be satisfied to know that the system is examining a large portion of the solution space, and that it is reasoning about it sensibly.

Our explanation approach is based on justifying a schedule through use of *examples*. To explain a schedule S , the explanation system generates a set of alternate solutions $\{A_0, \dots, A_i\}$ to the same scheduling problem, and then explains the superiority of S to each of the A_i s. The approach is designed to make the explanation compelling by selecting a diverse and relevant set of examples that covers as large a portion of the salient solution space as possible.

This approach is implemented in a prototype explanation module called EBE (Explanation By Example). EBE takes constraint solutions of the form produced by SRI's Calendar Manager [Berry et al. 2003] and produces explanations in both machine-readable and human-readable formats. It explains with examples selected to broadly cover the space of possible solutions, justifying the superiority of the system's solution for each example.

¹ That is not to say that the explanations will not also achieve other purposes secondarily. In particular, establishing trust typically entails enhancing the user's understanding of the problem and the system's solutions to it, and this enhancement is a secondary purpose of our explanations.

- Given:
 - A constraint problem $p=\{V, C, S, \geq\}$
 - V , a set of variables with collective domain D
 - C , a set of hard constraints on V
 - S , a set of soft constraints on V
 - $\geq: S \times S$, a preference relation on soft constraints
 - A solution $r=\{B, S_{met}, S_{unmet}\}$
 - $B: V \times D$, a set of variable bindings
 - $S_{met} \subseteq S$, the set of satisfied soft constraints
 - $S_{unmet} \subseteq S$, the set of unsatisfied soft constraints
- Algorithm:
 - GENERATE a collection of candidate examples
 - SELECT a final set of diverse examples E
 - For each example e in E
 - JUSTIFY r 's superiority to e

Figure 1: Problem and Top-level Algorithm for Example-based Explanation

2.1 Overview

The input and high-level approach of EBE is shown in Figure 1. EBE takes as input a constraint problem and a single solution produced by a constraint reasoner, hereafter referred to as the *solver*. The constraint problem consists of a set V of variables to be bound, a set C of hard constraints on V , that must be satisfied in any valid solution, a set S of soft constraints on V that are desirable but not required in a valid solution, and a relation \geq on S that partially orders the soft constraints by preference. Note that this problem formulation need not be the exact formulation accepted by the solver itself. For example, some constraint reasoners (including the Calendar Manager currently serving as a test bed for EBE) allow weights on the soft constraints and attempt to maximize the cumulative weight of the solution. EBE assumes only that the solver has some method of representing the importance of its soft constraints, and that those soft constraints can be partially ordered based on the system's own preference/importance scheme.

The solution consists of a set of variable bindings $B: V \times D$ (where D is the union of the domains of all variables in V), the subset $S_{met} \subseteq S$ of soft constraints satisfied in the solution, and the subset $S_{unmet} \subseteq S$ of soft constraints that are not satisfied in the solution. (Having both S_{met} and S_{unmet} in the solution is for convenience; since S is the disjoint union of S_{met} and S_{unmet} , one is easily calculable from the other.) The only assumption EBE makes about the solution is that the set of soft constraints met S_{met} is preferable to the corresponding set for any other valid solution, according to the solver's own solution ranking criteria. EBE assumes about how that ranking is calculated.

EBE's output is a collection of example alternate valid solutions to the same problem, and a justification of the solver's solutions superiority to each of those alternates. EBE creates this example-based explanation using a generate-select-justify cycle. It first *generates* a collection of candidate examples by creating a number of variants of the original problem and using the target system to solve them. Next, it *selects* the final set of examples, using a distance measure and greedy algorithm to make this set as diverse as possible. Finally, for each example in the final set, it *justifies* the superiority of the solution by identifying the salient differences between the two.

2.2 Generating Candidate Examples

The algorithm used to generate candidate examples constructs these examples by focusing on the solution's shortcomings—that is, those soft constraints the solver had to drop in order to produce the best solution. For each non-empty combination $_$ of dropped constraints, EBE creates a new problem by moving each constraint in $_$ from the soft constraints to the hard constraints, and then using the solver to solve the revised problem. These subsets are ordered from smallest to largest, and subsets containing high-priority dropped constraints are ordered before those containing lower-priority constraints. This ordering ensures that, in the next step (Selection), ties are broken in favor of simple examples, and secondarily in favor of those examples satisfying the most important constraints.

2.3 Selecting Diverse Examples

The example selection approach is a greedy algorithm, selecting at each iteration the example that maximizes a measure of *diversity* with respect to the current selected set. Our present measure of the diversity added by an example is the example's minimum distance from the members of the set. Our measure of the distance between two examples, in turn, is the number of variables for which the two solutions have different values.² This procedure ensures that the explanation is not presented with many examples that are only slight variants of one another.

2.4 Justifying Examples

After examples have been generated and selected, EBE uses each of them to justify the solution by arguing that the example suffers in comparison. It does this by focusing on differences between the solution and the example—any satisfied constraints or dropped constraints the two solutions have in common are ignored for the purposes of explanation. In particular, the justifications are based on comparing the *advantages* of each solution to the other:

² Because the examples considered for a given explanation have the same variables, there is no need to normalize the distance measure to achieve a fair comparison.

Definition 1 [Advantage Set] The *advantage set* of a solution r_1 to problem p over a solution r_2 to problem p is the set of p 's soft constraints met by r_1 but not by r_2 . That is,

$$\text{Advantages}(r_1, r_2) = S_{\text{met}}(r_1) \cap S_{\text{unmet}}(r_2)$$

The justification mechanism identifies the advantages of the example, and shows how the solution has a more important set of advantages.

EBE first attempts to find a one-to-one mapping between example advantages and a subset of the solution advantages, such that every solution advantage is preferred to its corresponding example advantage. If such a mapping exists, presenting it provides a compelling and concise justification of the superiority of the solution—showing how each advantage of the example is overwhelmed by a more important advantage of the solution.

Because we are not making any detailed assumptions about the solver's method of computing global preference, however, such a mapping may not exist. In this case, the justifier returns all the advantages of the solution and the example. This will presumably constitute less compelling raw material for an explanation than the point-by-point refutations mentioned earlier, but if the advantage sets are of manageable size, they may still be understandable and persuasive.

EBE outputs its explanations in a list format that could easily be converted into XML or some other format suitable to a natural language engine or another explanation display engine. We implemented a simple text generator that outputs an example-based explanation in a table. The text generator produces explanations at two different levels of detail, described below.

2.4.1 Concise Justification

The default mode for EBE is to justify each example using its concise format. In this format, the goal is to keep the size of the justification for each example as brief as possible. In particular, each justification includes only those solution advantages necessary to demonstrate the solution's superiority to the example. Any additional advantages of the solution are omitted.

Figure 2 shows an example of an explanation using concise justifications. The justification for each example shows

- The *primary* advantages of each example—i.e., the constraints that were “hardened” in the GENERATE step discussed above
- The *secondary* advantages of each example—i.e., any other soft constraints met in the example but not met in the solution
- The minimal collection of solution advantages necessary to demonstrate the superiority of the solution. If the solution advantages dominate the example advantages, this will be the highest-priority n solution advantages, where n is the

total number of example advantages. If not, it will be the complete set of solution advantages.

2.4.2 Verbose Justification

A user who wants more detail on a particular example can query EBE for a verbose justification of it. If the solution advantages dominate the example advantages, a verbose justification of an example differs from a concise justification in two ways:

1. It presents the mapping between each example advantage and the solution advantage that is preferred to it.
2. It presents each of the solution advantages, not just the minimal set in the concise justification.

2.5 Sample Explanation

EBE is implemented in Allegro Common Lisp, and has been tested with sample results from SRI's Calendar Manager. Figure 2 shows an example explanation of a simple meeting scheduling example. The objective is to schedule three (long) meetings during a day, with various hard constraints on attendee availability, room availability, room capacity, and so on, and soft constraints involving attendee preferences, times preferences, room preferences, and so on.

The solver produces a solution that schedules a large staff meeting and a small project meeting in parallel in the afternoon, and a mid-sized project meeting in the morning. This solution entails a number of dropped soft constraints; of interest here are that a few people are unable to attend the staff meeting, and the smaller project meeting is not assigned to its preferred room.

THE BEST SOLUTION THAT MEETS THESE CRITERIA:	...ALSO MEETS THESE CRITERIA:	...BUT DOESN'T MEET THESE MORE IMPORTANT CRITERIA:
(INCLUDES STAFF-MTG-04006.ATTENDEES "Ian Harrison")	(INCLUDES STAFF-MTG-04006.ATTENDEES "Michael Wolverton")	(INCLUDES STAFF-MTG-04006.ATTENDEES "Tom Garvey") (!OVERLAPS STAFF-MTG-04006.START-END "[1200, 1300]")
(INCLUDES STAFF-MTG-04006.ATTENDEES "Michael Wolverton")		(INCLUDES P11590-TRANSITIONTF-MTG-04012.ATTENDEES "Michael Wolverton")
(= P11590-TRANSITIONTF-MTG-04012.LOCATION "EJ228")		(INCLUDES P11590-PRJ-MTG-04003.ATTENDEES "Andres Rodriguez")

Figure 2: Example Explanation in Table Format

For the explanation in Figure 2, EBE's settings limited it to three examples. Of the multiple examples it generated in the first step, it selected three that produced distinctly different solutions by its diversity measure. Among those solutions that add equal diversity, it preferred ones whose problem was closest to the original. This led it to choose examples for three of the dropped constraints in the solution. All other examples produced solutions very similar to or, in some cases, identical to one of those three.

The simple text generator we implemented displays solution and example advantages in a Lisp-like format. An English translation of the output of Figure 2 would be something like:

The best schedule that allows Ian Harrison to attend the staff meeting also allows Michael Wolverton to attend the staff meeting. However, it suffers from the more important disadvantages that (1) Tom Garvey cannot attend the staff meeting, and (2) the staff meeting overlaps the 1200-1300 lunch hour.

The best schedule that allows Michael Wolverton to attend the staff meeting suffers from the more important disadvantage that Michael Wolverton cannot attend the P11590 Transition Task Force meeting.

The best schedule that locates the P11590 Transition Task Force meeting in room EJ228 suffers from the more important disadvantage that Andres Rodriguez cannot attend the P11590 project meeting.

2.6 Discussion

Using examples to drive explanation generation is an appealing alternative to—or supplement to—traditional chain-of-reasoning explanation methods. EBE helps show the promise of the approach, but it clearly represents only an early step toward a full-fledged

explanation-by-example system. There are many issues to address in this area. Among them are

- Better and more detailed justifications. EBE currently takes a “cause-and-effect” approach to justifying the solution with respect to an example, showing just the advantages of the example and the disadvantages it ultimately created. However, in many cases the justifications leave unclear the reason *why* the advantages led to the disadvantages. For instance, in the last example of Figure 2, it probably will not be obvious to most users how changing a meeting’s location leads to Andres Rodriguez being unable to attend a different meeting. EBE needs a method of connecting the dots. In particular, it needs to use both the solution and the example to show how a change in one element of the solution leads to a particular shortcoming in the example. Combining example-based techniques with more traditional chain-of-reasoning constraint explanation methods (e.g., [Sqalli & Freuder 96]) seems promising here.
- An improved model of explanation coverage. Ideally, examples should be selected based on diversity and salience. Diversity is important to prevent the explanation from including multiple examples that convey nearly identical information. Salience is important to keep the explanation focused on the most important questions or doubts the user may have about the reasoning that led to the solution. The current EBE model attaches great importance to diversity, but probably underemphasizes salience. We need better methods for anticipating user information needs, possibly even user-specific methods based on user models.
- Improved presentation and evaluation. End-user explanations obviously require more user-friendly formats than the Lisp forms shown in Figure 2. We need methods of presenting example attributes (e.g., advantages and disadvantages) in understandable natural language and/or in graphical illustrations. We also need to evaluate explanation techniques by measuring acceptance, understandability, and task performance with human users.

3 Qualitative Plan Summarization and Comparison

The motivation for our work on qualitative plan summarization and comparison was to provide a human planner with tools that help in understanding the key elements of an individual plan and identify important differences among alternative plans. Two principles drove our work in this area: *domain independence* and *semantic grounding*.

The requirement for *domain-independent* methods was motivated by a desire to develop techniques that could be applied across a broad range of problems. In particular, we wanted to avoid reliance on domain-specific algorithms or bodies of knowledge that would limit the applicability of our tools. One problem with general-purpose methods is that their generality often comes at the cost of depth. This tradeoff applies in the design that we have chosen, in that more discriminating tools could be developed on a domain-by-domain basis that provide deeper summarization and comparison capabilities. However, we feel that our approach can provide significant value, particularly in the initial design stages of formulating a plan, when the user is looking to understand the high-level tradeoffs among alternative candidate solutions. In particular, our tools would be helpful to a human planner who would like to understand better the range of options available within a large, complex solution space. If desired, more detailed and specialized evaluation tools could be used to perform final evaluations on a short list of candidate plans.

Our requirement for semantic grounding stems from a belief that semantic methods are essential for obtaining meaningful insight into plans. In contrast, much of the work on explaining plans to date has been tightly linked to syntactic characteristics of a plan's structure or the underlying reasoning processes used to generate it. For example, the work by [Young 1999] on plan summarization rates the importance of an action within a plan by counting the number of its incoming causal links; only actions with certain numbers of links are included in the plan summary. One serious problem with such approaches is that a plan structure and the reasoning processes used to create it do not generally match the user's conceptualization of the domain.

To provide a semantic grounding for our summarization and comparison methods, we exploit a *domain metatheory*. The domain metatheory is an abstract characterization of a planning domain that specifies important semantic differences among operators, planning variables, and instances. This abstraction provides the means to describe plans and their components in high-level, semantically meaningful terms. The concept of the domain metatheory was introduced originally to provide a language that would enable a user to define *advice* for directing a planning system without having to acquire detailed knowledge of its internal workings or its constituent knowledge [Myers 1996]. Similarly, we believe that it can be used as the basis for communicating explanatory information in a way that more closely aligns with the user's conceptualization of the domain.

3.1 Overview

Our approach to plan summarization and comparison revolves around a suite of techniques that look for regularities or interesting exceptions relative to key aspects of the domain metatheory. For example, a metatheory *role* corresponds to an important actor or object within a plan. In comparing two plans, one interesting dimension to consider is whether the plans fill key roles in different ways (e.g., *role differencing*). For example, two plans may be similar in structure but one uses the front camera for imaging while the other uses the rear camera. A similar technique could be applied for metatheory *features*, which correspond to semantic attributes of an operator, such as the associated level of risk or cost. Feature differencing could be used, for example, to note that one plan employs a high-risk traversal maneuver while the other plan uses a low-risk approach. Similar techniques for summarization can also be defined that summarize trends relative to the use of a particular role or feature within a single plan. For example, it could be useful to note that the only action in a given plan that is considered high risk is the traversal maneuver; otherwise, the plan is low risk.

The plans to which our summarization and comparison methods can be applied could be generated in a number of ways. First, automated planning systems (e.g., SIPE-2 [Frank & Jónsson 2003; Wilkins 1993]) could be used to create a range of alternatives, possibly using some biasing techniques to generate options that would be expected to differ in interesting ways. Second, the plans could be developed using some form of plan authoring tool that enables the user to create a range of options interactively (e.g., PASSAT [Ai-Chang et al. 2003; Myers et al. 2002]). Third, some form of semi-automated method (e.g., plan sketching [Myers et al. 2003]) could be used in which the user outlines key aspects of the plan but automated tools are used to flesh out the plan to alternative solutions.

Our work on qualitative plan summarization and comparison makes three contributions. First, we defined a domain-independent framework for summarization and comparison that is grounded in the use of a domain metatheory. Second, we implemented a suite of tools grounded in this theory within the PASSAT mixed-initiative planning framework [Myers et al. 2002]. These tools support (a) summarization of an individual plan, (b) comparison of pairs of plans, and (c) analysis of a collection of plans. Finally, we showed the potential for these methods in helping a user understand individual and collections of plans by evaluating the tools within an extensive domain for special operations planning.

3.2 Technical Background

Our plan summarization and comparison work assumes a *hierarchical task network* (HTN) paradigm for representing plans similar to that described in [Erol et al. 1994]. Most large-scale, realistic planning applications to date have made use of the HTN paradigm, as it has two important advantages over classical planning methods. First, task networks can be used to encode known strategies for task decomposition, thus eliminating the need for the planner to ‘discover’ such strategies on its own during plan generation. The use of such precompiled strategies generally leads to greatly improved performance over

classical techniques. Second, the structure of planning knowledge within the HTN framework often better matches user intuitions about the domain, both in terms of encoding specific problem-solving strategies explicitly, and the hierarchical breakdown of task structures.

Within the HTN framework, we assume a type hierarchy for terms within the domain model. Thus, each domain individual has an associated type *TYPE(i)*, and it is possible to identify the most specific supertype that encompasses a set of terms.

3.3 Domain Metatheory

A standard HTN domain theory consists of four basic types of element: *individuals* corresponding to real or abstract objects in the domain, *relations* that describe characteristics of the world, *tasks* to be achieved, and *templates* that describe available means for achieving tasks. (Templates are alternatively referred to as methods or operators in the literature.)

A *domain metatheory* defines semantic properties for domain theory elements that enable high-level descriptions of activity that abstract from the syntactic details of the domain knowledge. The concept of a domain metatheory for planning was introduced in [Myers 1996] as a way of providing a semantically grounded language in which a user could give *advice* to a planning system. Advice, which describes high-level characteristics of desired solutions, can be operationalized into structures and mechanisms that guide an automated planning system at runtime to make choices that best match the advice. Subsequently, the metatheory was also used as the basis for generating qualitatively different plans, by using structure within the metatheory to direct a planning system toward solutions with distinct semantic traits [Myers & Lee 1999].

Our plan summarization and comparison techniques can be viewed, in some sense, as a kind of ‘inverse’ of the advice-giving process. In particular, we take a solution to a planning problem and try to extract from it interesting characteristics that can be described in terms of the metatheory. The value of the metatheory for this type of extraction is that it provides a semantic framework for guiding the choice of concepts used in summarizing and comparing plans. As such, the summarizations and comparisons are grounded in semantically significant concepts rather than syntactic constructs whose meaning or import is unclear.

The metatheory that we use for plan summarization and comparison is similar to that introduced for the work on advisable planning. To support our work on summarization and comparison, however, we introduced a few extensions and refinements to the original conceptualization that provide a somewhat richer and more structured framework. The main metatheoretic concepts that we make use of are *template features*, *task features*, and *roles*.

3.3.1 Template Features

A *template feature* designates an intrinsic characteristic of a template that distinguishes it from other templates that could be applied to the same task. So, for example, among templates that could be applied to a transportation task, there may be one that is *fast* but *expensive* with a second that is *slow* but *cheap*. Although the two templates are functionally equivalent in that they accomplish the same task, their intrinsic characteristics differ significantly. Template features provide the means to distinguish among such functionally equivalent alternatives.

We model template features in terms of a *feature category* (e.g., *cost*) and a feature value (e.g., *expensive*). Features values are drawn from a predefined set of values that constitute the *domain* of the feature category. For this work, we require that the domain for a template feature be totally ordered (that needn't be true in general). Because our interests lie with qualitative comparison of plans, we make use of ordinal domains defined in terms of a small number of qualitative feature values. In the metatheory for the special operations domain, we define the domain for each category to be the ordered collection of values [*low medium high*]. A given template feature may be declared for multiple templates within a domain, possibly with the same or different feature values. So, for example, there may be multiple templates whose *cost* is considered *expensive* relative to other available options.

We say that a *template feature* f with value v occurs in plan P iff there is some template T applied to a task t in P such that T has the feature f with value v . In general, a plan may have multiple occurrences of a given template feature that cut across a broad range of templates used to accomplish different types of tasks. Different occurrences may have different values associated with them, although duplication of values is also possible. The term $TemplateFeatureInsts(f,P)$ denotes the collection of values for occurrences of template feature f in plan P , including duplications for multiple feature/value occurrences.

The utility of template features for plan summarization and comparison is that they provide the means to identify, abstract and contrast important evaluational properties of different strategies, such as speed or cost. In particular, template features can be used as a kind of 'quick and dirty' proxy for deeper and more significant evaluations of a plan.

3.3.2 Task Features

Task features correspond to semantic attributes for a task. For example, there may be several types of reconnaissance task: satellite reconnaissance, ground reconnaissance, and aircraft reconnaissance. Each of these tasks can be labeled as having the feature *RECON*, thus providing a mechanism for abstracting over that set of tasks. (A similar sort of semantic grouping could be achieved through the use of a class hierarchy for tasks.) Task features can be viewed as a restricted type of feature whose domain consists of the values [*false true*].

3.3.3 Roles

A *role* describes a capacity in which a domain object is used within a template or task; it maps to an individual template or task variable. For instance, a template for transporting materials may contain variables `location.1` and `location.2`, with the former corresponding to the `START` role and the latter the `DESTINATION` role for the move. Roles provide a semantic basis for describing the use of individuals within templates and tasks that abstracts from the details of specific variable names. Roles also provide the means to reference a collection of semantically linked variables that span different templates and tasks (i.e., `START` roles may occur in multiple templates and tasks).

We say that a *role* r with fill v occurs in plan P iff either:

- there is some task $t(a_1, \dots, a_n)$ in P where $t(a_1, \dots, a_n)$ is of type $TaskType_k$, argument i for task type $TaskType_k$ is declared to represent the role r , and $a_i = v$,
or
- some template T with role r defined for local variable var_i is applied to a task $t(a_1, \dots, a_n)$ in P and var_i is bound to value v .

In general, a plan may have multiple occurrences of a given role with different or identical fills. The term $RoleFills(r, P)$ denotes the collection of values that occur as fills for role r in plan P .

3.4 Experimental Framework

We evaluated the effectiveness of our plan comparison techniques on a suite of nine test plans from a *special operations forces* (SOF) domain. This domain was created with assistance from members of the special operations community, as part of an earlier project focused on developing mixed-initiative planning technology within the PASSAT system [Myers et al. 2002]. The SOF domain constitutes a sizable and rich test environment for evaluating our work on plan comparison: the base-level domain contains 65 predicates modeling key world properties, more than 100 tasks, and more than 50 templates spanning a hierarchy of five distinct abstraction layers.

The original SOF domain included a limited metatheory designed to showcase advice-taking within PASSAT. For this project, we extended the domain to include a fairly comprehensive metatheory that includes 13 template features, 12 task features, and more than 75 roles. Figure 3 summarizes the feature categories defined within the SOF domain. The task features employ the domain of values `[false true]`; the template features employ the domain `[low medium high]`.

The test plans address the high-level task of extracting a set of hostages held by a guerilla team in an urban environment. More specifically, this task requires rescuing a set of hostages being kept at Mogadishu-Town-Hall using forces based at Riyadh Airport, and then evacuating the hostages to a safe haven at Riyadh Stadium. This top-level task is represented as

(Rescue-Hostage Mogadishu-Town-Hall Riyadh-Airport Riyadh-Stadium)

Template Features	Task Features
BLUE-CASUALTY-RISK COLLATERAL-DAMAGE COORDINATION-COMPLEXITY COVERTNESS DURABILITY FORCE-FATIGUE FORCE-FOOTPRINT FORCE-INTEGRITY INFORMATION-QUALITY LANDING-ZONE-PREPARATION-DEMAND ROBUSTNESS SPEED VULNERABILITY-GROUND-FIRE	CSAR-SUPPORT DIVERSION EVACUATION FIRE-SUPPORT PARACHUTE RECON REFUELING RESCUE RESCUE-AND-RECOVER SEAD SECURITY SUPPORT

Figure 3. Task Feature Categories for the SOF Rescue Domain

The SOF base-level domain includes a number of templates that reflect different strategies for rescuing the hostages. Variations among solutions result from three sources. The first is whether the plan contains certain strategic and tactical types of activities; depending on a given situation, the commander may or may not decide to include such activities within the plan. For example, while it is not necessary to create one or more diversions to distract the guerillas from the main activities, doing so may be desirable in some circumstances. The second relates to the selection of types of resources to be used to accomplish tasks. In some cases, for example, it may be appropriate to use satellites to gather intelligence information while in others it may be preferable to rely on ground forces. The third relates to decisions about key parameters within a plan, such as where to establish a forward base or the drop point for inserting the assault team.

Plan Identifier	Description
<i>tiny-plan-a</i>	Very simple plan without security or support
<i>tiny-plan-b</i>	Variant on <i>tiny-plan-a</i> that uses a different type of rescue force
<i>small-plan-a</i>	Basic solution that includes reconnaissance and combat search and rescue
<i>small-plan-b</i>	Variant on <i>small-plan-a</i> that uses the same high-level strategy but differs in the lower-level realization of parts of it
<i>medium-plan-a</i>	Broadly similar to the small plans but involves refueling
<i>medium-plan-b</i>	Broadly similar to the small plans but with SEAD activities
<i>large-plan-a</i>	Extensive plan with significant reconnaissance and support activities as well as a diversion from the main assault
<i>large-plan-b</i>	Variant on <i>large-plan-a</i> that provides increased fire support and SEAD
<i>large-plan-c</i>	Variant on <i>large-plan-a</i> with a different style of diversion

Figure 4. Summary of Test Plans

Figure 4 summarizes the nine test plans used in our evaluation. These plans were created by the author of the SOF domain knowledge, through a combination of manual and semi-

automated methods within the PASSAT system.³ The author was asked to create a core set of plans reflecting a representative set of strategic alternatives that a SOF commander might consider. In addition, he was asked to create variants on some of the core plans by making a few key strategic changes that might correspond to handling contingencies in different ways. Given that variants of this type are commonly made in practice, we were interested in determining how well our plan comparison techniques would be able to recognize the differences underlying such variants.⁴

3.5 Plan Summarization

The roles and features of the metatheory provide a semantic basis for summarizing key properties of a plan. In particular, a description of how a plan fills its roles, as well as the features that it possesses, can provide valuable insight into the structure and overall strengths and weaknesses of a plan.

Task Features Consider first task features, which provide a semantic summary of key activities within a plan. A listing of the task features for a given plan can inform the user that a given plan does or does not contain critical activities such as reconnaissance or fire support.

Template Features Template features provide a different perspective on a plan, as they characterize plan characteristics that have more of an evaluational nature (e.g., cost, speed). One important aspect of template features is that they can be applied in multiple contexts within a plan, with different occurrences yielding different values. This variation reflects the fact that, for example, a given plan may use an inexpensive reconnaissance operation but an expensive rescue strategy.

To enable summarization of the property represented by a template feature, we introduce the concept of the overall *template feature value* for a feature f and plan P , denoted by $TemplateFeatureValue(f,P)$. This value is defined to be the average of the values in $TemplatFeatureInsts(f,P)$, which (as defined above) is the collection of role fill values for all occurrences of feature f within plan P .

The use of a qualitative set of values for the feature domain introduces a slight complication in defining $TemplateFeatureValue(f,P)$, as it is necessary to support qualitative equivalents to operations such as addition and division. To this end, we require for each template feature f the definition of a surjective, order-preserving mapping θ_f from a designated interval of the reals $Interval(f)$ to the domain for the feature f :

$$\theta_f: Interval(f) \rightarrow Domain(f)$$

³ The author of the plans, although an AI researcher, was not involved in the design of the plan summarization and comparison techniques developed on this project.

⁴ These plans are more limited in scope and diversity than what one would expect from a commander free to formulate plans on his own, due to limitations on the scope of problem-solving knowledge within the planning tool.

Variation in the ‘qualitative closeness’ of values in $\text{Domain}(f)$ can be achieved by appropriate definitions of θ_f .⁵

Definition 1 [Template Feature Value] The *template feature value* for a feature f and plan P , denoted by $\text{TemplateFeatureValue}(f, P)$, is defined by

$$\text{TemplateFeatureValue}(f, P) = \theta_f \left(\frac{\sum_{v \in \text{TemplateFeatureInsts}(f, P)} \theta_f^{-1}(v)}{|\text{TemplateFeatureInsts}(f, P)|} \right)$$

Roles A description of how roles are filled within a plan can provide a concise summary of what resources are used and how, as well as key parameters to a plan (e.g., the choice of location for a forward base). Furthermore, it is possible to search for patterns in the filling of roles. So, for example, it may be useful to know that only satellites are used as reconnaissance assets, or that all transport of troops is through the use of helicopters of a particular type. We refer to such patterns as *uniformities* in the filling of roles. Here, we define two specific types of uniformity for role fills oriented around *values* and *types*.

Definition 2 [Value Uniformity in Role Fills] A plan P *uniformly fills a role r with value c* iff $|\text{RoleFills}(r, P)| > 1$ and $v \in \text{RoleFills}(r, P)$ implies that $v = c$.

Type uniformity depends on the declaration of a type $\text{TYPE}(r)$ for a given role r that delimits the set of possible fills for r ; in other words, all fills for role r must be of type $\text{TYPE}(r)$. Type uniformity becomes interesting when there is a proper subtype of $\text{TYPE}(r)$ that generalizes all fills for a given role. For example, it can be interesting to note that only satellites are used for reconnaissance, when other types of assets (e.g., ground forces) are possible.

Definition 2 [Type Uniformity in Role Fills] A plan P *uniformly fills a role r with a type T* iff $|\text{RoleFills}(r, P)| > 1$, T is a proper subtype of $\text{TYPE}(r)$, and every fill value $v \in \text{RoleFills}(r, P)$ is of type T .

Value and type uniformity for roles constitute generic, domain-independent mechanisms for generalizing a collection of role fills. For a given domain, it may be appropriate to introduce additional generalization mechanisms. For example, in domains where locations play a significant role, it might be useful to generalize based on geographic proximity, or co-location within some designated geographic unit (e.g., all air assets are pulled from bases in the same region).

⁵The SOF domain metatheory makes use of the domain [low medium high], the interval [0 ... 1], and the common mapping function $\theta: [0 .. 1] \rightarrow \{\text{low medium high}\}$ for every template feature in the domain. Furthermore, θ^{-1} is distributed linearly in the range [0 ..1], with low mapping to 0, medium to 0.5 and high to 1.

3.5.1 Sample Plan Summary

To illustrate the value of metatheory-based plan summarization, consider the summary of the test plan *medium-plan-b* shown in Figure 5.⁶ As can be seen, the plan is sufficiently complex that its key strategic elements and attributes are not readily apparent. Rather, some form of analysis tool (such as metatheoretic summarization) is required to understand the plan.

Figure 6 summarizes the task features within this plan, while Figure 7 summarizes the template features and their normalized values. Figure 8 summarizes certain key role fills for the plan.

The summary of task features in Figure 6 makes it easy to identify the key strategic elements of the plan. The features **RESCUE-AND-RECOVER** and **RESCUE** derive from the fact that the plan describes a rescue-and-recover operation; these features are common to every plan in the test suite. At a lower level, we can see that this particular solution includes components for combat search and rescue support (**CSAR-SUPPORT**), fire support (**FIRE-SUPPORT**), reconnaissance (**RECON**), and suppression of enemy air defenses (**SEAD**). As it turns out, these components are all optional in the sense that not every solution is required to contain them.

The template features in Figure 7 summarize key evaluational qualities of the plan. Desirable qualities include the fact that the expected quality of information underlying the plan is high (**INFORMATION-QUALITY** has value **high**), while expected collateral damage is low (**COLLATERAL-DAMAGE** has value **low**). On the negative side, the overall solution is highly vulnerable to ground fire (**VULNERABILITY-GROUND-FIRE** has value **high**).

More than 30 roles occur in the plan *medium-plan-b*, some of which have multiple fills. Typically, a user would not choose to view all roles and their fills at once. Rather, at a given point in time he would be interested in knowing about a subset of these roles as he focuses on certain aspects of the plan. So, for example, a user interested in understanding the high-level strategy of a plan may concentrate on a subset of roles related to key strategic decisions, while a user interested in asset usage may concentrate on roles related to resource utilization.

⁶ Figure 5 presents a task decomposition view of the plan that highlights the hierarchical organization of its constituent activities. For simplicity, temporal sequencing information among activities (which is less important for understanding the plan's structure and design) has been omitted.

- * (Rescue-Hostage Mogadishu-Town-Hall Riyadh-Airport Riyadh-Stadium)
 - * (Rescue-And-Recover Riyadh-Airport Mogadishu-Town-Hall Riyadh-Stadium)
 - * (Recon Mogadishu-Town-Hall)
 - * (Infiltrate Green-Oda-2 Ankara-Airport Mogadishu-Town-Hall)
 - * (Produce-Landing-Plan Mh-60-G-Pave-Hawk-2)
 - * (Produce-Air-Movement-Plan Mh-60-G-Pave-Hawk-2)
 - * (Produce-Loading-Plan Green-Oda-2)
 - * (Produce-Aircraft-Bump-Plan Green-Oda-2)
 - * (Load Green-Oda-2 Mh-60-G-Pave-Hawk-2)
 - * (Fly Mh-60-G-Pave-Hawk-2 Ankara-Airport Mogadishu-Stadium)
 - * (Drop Green-Oda-2 Mh-60-G-Pave-Hawk-2 Mogadishu-Town-Hall)
 - * (Depart Mh-60-G-Pave-Hawk-2 Mogadishu-Stadium)
 - * (Establish-Observation-Post Green-Oda-2 Mogadishu-Town-Hall)
 - * (Exfiltrate Green-Oda-2 Mogadishu-Town-Hall Mogadishu-Building4)
 - * (Fly Uh-60a-2 Mogadishu-Town-Hall Mogadishu-Building3)
 - * (Load Green-Oda-2 Uh-60a-2)
 - * (Depart Uh-60a-2 Mogadishu-Building3)
 - * (Provide-Fire-Support Mogadishu-Town-Hall)
 - * (Take-Off Ch-53e-Super-Stallion-1 Addis-Ababa-Airport)
 - * (Fly Ch-53e-Super-Stallion-1 Addis-Ababa-Airport Mogadishu-Town-Hall)
 - * (Place-On-Station-Fire-Support Ch-53e-Super-Stallion-1 Mogadishu-Town-Hall)
 - * (Fly Ch-53e-Super-Stallion-1 Mogadishu-Town-Hall Addis Airport)
 - * (Land-At Ch-53e-Super-Stallion-1 Addis-Ababa-Airport)
 - * (Provide-Csar-Coverage Csar-Team-2 Mogadishu-Town-Hall)
 - * (Prepare Csar-C1-A)
 - * (Take-Off Csar-C1-A Baidoa-Stadium)
 - * (Fly Csar-C1-A Baidoa-Stadium Mogadishu-Town-Hall)
 - * (On-Station Csar-C1-A Mogadishu-Town-Hall)
 - * (Provide-Fire-Support Mogadishu-Town-Hall)
 - * (Take-Off Ah-100-1 Balikesir-Stadium)
 - * (Fly Ah-100-1 Balikesir-Stadium Mogadishu-Town-Hall)
 - * (Place-On-Station-Fire-Support Ah-100-1 Mogadishu-Town-Hall)
 - * (Fly Ah-100-1 Mogadishu-Town-Hall Balikesir-Stadium)
 - * (Land-At Ah-100-1 Balikesir-Stadium)
 - * (Provide-Sead Sead-1 Ad-Dammam-Stadium Mogadishu-Town-Hall)
 - * (Prepare Sead-1)
 - * (Take-Off Sead-1 Ad-Dammam-Stadium)
 - * (Fly-To Sead-1 Mogadishu-Town-Hall)
 - * (Infiltrate Orange-Oda-1 Riyadh-Airport Mogadishu-Town-Hall)
 - * (Fly-Commercial Aa7864 Orange-Oda-1 Riyadh-Airport Mogadishu-Town-Hall)
 - * (Storm Orange-Oda-1 Mogadishu-Town-Hall)
 - * (Exfiltrate Orange-Oda-1 Mogadishu-Town-Hall Riyadh-Stadium)
 - * (Fly-Commercial Aa201 Orange-Oda-1 Mogadishu-Town-Hall Riya Stadium)
 - * (Provide-Fire-Support Mogadishu-Town-Hall) [Fire-Support-Naval]
 - * (Station Yorktown Mogadishu-Town-Hall)
 - * (Provide-Csar-Coverage Csar-Team-2 Mogadishu-Town-Hall)
 - * (Prepare Uh-601-1)
 - * (Take-Off Uh-601-1 Bihac-Stadium)
 - * (Fly Uh-601-1 Bihac-Stadium Mogadishu-Town-Hall)
 - * (On-Station Uh-601-1 Mogadishu-Town-Hall)

Figure 5. Task Decomposition View of the Plan *medium-plan-b*

Task Feature	Present
CSAR-SUPPORT	Y

DIVERSION	N
EVACUATION	N
FIRE-SUPPORT	Y
PARACHUTE	N
RECON	Y
REFUELING	N
RESCUE	Y
RESCUE-AND-RECOVER	Y
SEAD	Y
SECURITY	N
SUPPORT	Y

Figure 6. Summary of Task Features for Plan *medium-plan-b*

Template Feature	Plan Value
BLUE-CASUALTY-RISK	medium
COLLATERAL-DAMAGE	low
COORDINATION-COMPLEXITY	medium
COVERTNESS	medium
DURABILITY	low
FORCE-FATIGUE	medium
FORCE-FOOTPRINT	medium
FORCE-INTEGRITY	medium
INFORMATION-QUALITY	high
LANDING-ZONE-PREP-DEMAND	low
ROBUSTNESS	medium
SPEED	medium
VULNERABILITY-GROUND-FIRE	high

Figure 7. Summary of Template Features for Plan *medium-plan-b*

Roles Related to Force Usage

Role	Fill Values
ASSAULT-FORCE	green-oda-2 orange-oda-1
FORCE	orange-oda-1 (2) green-oda-2 (2)
OBSERVATION-FORCE	green-oda-2

Roles Related to Asset Usage

Role	Fill Values
AIR-ASSET	uh-601-1 sead-1 ah-100-1 csar-cl-a ch-53e-super-stallion-1 uh-60a-2 mh-60-g-pave-hawk-2
CSAR-HELICOPTER	uh-601-1 csar-cl-a
FIRE-SUPPORT-ASSET	yorktown ah-100-1 ch-53e-super-stallion-1
RECON-ASSET	green-oda-2
SEAD-AIRCRAFT	sead-1 (2)
TRANSPORT-ASSET	sead-1 uh-60a-2 mh-60-g-pave-hawk-2

Roles Related to Strategic Decisions

Role	Fill Values
ASSAULT-LOCATION	mogadishu-town-hall
CSAR-LOCATION	mogadishu-town-hall (2)
EXFIL-POINT	mogadishu-town-hall mogadishu-building4
FIRE-SUPPORT-LOCATION	mogadishu-town-hall (5)
FORWARD-BASE	ankara-airport
INFILTRATION-START	riyadh-airport ankara-airport
OBSERVATION-LOCATION	mogadishu-town-hall

Figure 8. Summary of Selected Roles for Plan *medium-plan-b*

Figure 8 displays three sets of role fills from the plan *medium-plan-b*; these sets were chosen as representative ‘views’ of the plan that might be of interest to the user. For fill values that occurred more than once for a given role, the number of occurrences is noted in parentheses.

The first set consists of roles related to force usage. This summary makes it easy to see that only Green and Orange teams are used in the plan⁷; both are used in assault roles while the Green team is also used in a reconnaissance capacity as an observation force. The second set shows roles related to asset usage, broken down by asset type in some

⁷ The color used in the name of a force is highly significant: different colors denote units from different services that come with skills and capabilities. For the sake of brevity, we omit detailed descriptions of the qualities associated with the different force colors.

cases (e.g., AIR-ASSET) and functional roles (e.g., FIRE-SUPPORT-ASSET, TRANSPORT-ASSET). The third set shows select roles corresponding to strategic decisions within the plan, such as the location for the assault and reconnaissance efforts (i.e., ASSAULT-LOCATION, OBSERVATION-LOCATION).

Figure 9 summarizes value-based and type-based role uniformities for the plan *medium-plan-b*. For value-based uniformity, the summary indicates the role, the fill value, and the number of occurrences. For type-based uniformity, the summary indicates the role, its type, the types of the fill values, and the most specific type that generalizes the fills. The information on type-based uniformity is particularly useful in this case as it highlights the exclusive use of air assets for many key functional roles within the plan.

<i>Value-based Role Uniformity</i>			
Role	Value	Count	
CSAR-LOCATION	mogadishu-town-hall	2	
FIRE-SUPPORT-LOCATION	mogadishu-town-hall	5	
FORWARD-POINT	riyadh-airport	3	
INFILTRATION-DESTINATION	mogadishu-town-hall	2	
SEAD-AIRCRAFT	sead-1	2	

<i>Type-based Role Uniformity</i>			
Role	Role Type	Fill Types	Min. SuperType
EXFIL-ASSET	ASSET	COMMERCIAL-FLIGHT HELICOPTER	AIR-ASSET
FIRE-SUPPORT-ASSET	PHYSICAL-ENTITY	WARSHIP HELICOPTER HELICOPTER	ASSET
INFIL-ASSET	ASSET	COMMERCIAL-FLIGHT HELICOPTER	AIR-ASSET
TRANSPORT-ASSET	ASSET	SEAD-AIRCRAFT HELICOPTER HELICOPTER	AIR-ASSET

Figure 9. Role Uniformities in Plan *medium-plan-b*

3.6 Comparing Two Plans

Our approach to comparing plans is grounded in two techniques: *feature differencing* and *role differencing*. These techniques can be useful in terms of both identifying subtle variations in similar plans and understanding larger differences in more varied plans.

3.6.1 Feature Differencing

As noted above, features within a plan correspond to high-level semantic characteristics of tasks (for task features) and strategy or evaluational qualities (for template features) for a plan.

Task features provide a semantic summary of key activities within a plan. Task feature differencing, which involves a comparison of task features within two plans, provides a

quick snapshot of how the two plans differ in their key task types. This type of capability can enable a user to easily see, for example, that one plan contains reconnaissance capabilities while another does not.

Template feature differencing involves comparing the normalized template feature values for two different plans in order to identify significant variations. This form of differencing makes it easy to see, for example, that one plan trades risk for increased complexity relative to another plan.

3.6.2 Role Differencing

Role differencing looks at variabilities in how two plans fill their roles. This type of comparison can shed insight on key differences in both strategic decisions (e.g., *Where are the hostages to assemble?*) and resource usage (e.g., *What types of reconnaissance asset are used?*).

Disjoint: $V_1 _ V_2 = \{\}$

Different single-valued: $V_1 _ V_2 = \{\} \wedge |V_1|=|V_2|=1$

Disjoint types: $MinSupertype(V_1) \neq MinSupertype(V_2)$
 $\wedge MinSupertype(V_1) \not\subset MinSupertype(V_2)$
 $\wedge MinSupertype(V_2) \not\subset MinSupertype(V_1)$

Disjoint multivalued: $V_1 _ V_2 = \{\} \wedge (|V_1|>1 \vee |V_2|>1)$

Overlapping: $V_1 _ V_2 \neq \{\}$

Restricted subtype: $MinSupertype(V_1) \subset MinSupertype(V_2) \vee$
 $MinSupertype(V_2) \subset MinSupertype(V_1)$

Restricted subset: $V_1 \subset V_2$

Unstructured overlap: $V_1 \neq V_2 \wedge V_1 _ V_2 \neq \{\} \wedge V_1 \not\subset V_2 \wedge V_2 \not\subset V_1$

Figure 10. Categories of Role-fill Differences

Figure 10 summarizes key ways in which the fill values for a given role in two plans can differ. There, V_1 and V_2 designate sets of fill values for a given role from which duplicates have been removed. Further, it is assumed that $V_1 \neq V_2$ and that both V_1 and V_2 are non-empty. The notation $MinSupertype(V)$ denotes the most specific type that contains all of the values in V .

The first three entries cover situations where V_1 and V_2 are disjoint; the last three entries cover situations where V_1 and V_2 overlap. Within these two collections, the categories are ordered with respect to the relative strength of the requirement: categories that appear higher in the list impose more stringent conditions than categories lower in the list.

The category *different single-valued* can be useful for identifying differences in key individual decisions for a plan. Here, we provide two examples drawn from our suite of test plans.

- For the role ASSAULT-FORCE, the plan *tiny-plan-a* uses Orange-ODA-2 while the plan *tiny-plan-b* uses Green-ODA-1. This difference is significant, as noted above, because the orange and green forces have significantly different core capabilities.
- For the role RECON-ASSET (which designates the type of resource to be used for reconnaissance), the plan *medium-plan-a* uses satellite-1 while the plan *tiny-plan-b* uses Green-ODA-2. Because the nature of the intelligence information that can be obtained with these two types of assets is markedly different, this distinction in the two plans is important.

The category *disjoint types* requires both that the most specific supertype of the role-fill values in the two plans be different, and that neither is a subtype of the other. As an example, the plan *small-plan-a* uses helicopters of type CSAR-HELICOPTER-CLASS-1 exclusively for combat search and rescue while the plan *large-plan-b* uses helicopters of type CSAR-HELICOPTER-CLASS-2. The category *disjoint multivalued* defines an even weaker condition, requiring only that the fill values for the two plans be different.

The category *different single-valued*, although just a special case of *disjoint types* and *disjoint multivalued*, is useful to distinguish. That is especially true when the role appears exactly one within each of the two plans being compared; such a role often designates some critical parameter choice for a plan (e.g., the role ASSAULT-FORCE in the example above).

For overlapping values, the strongest condition is *restricted subtype*, which indicates that the most specific supertype of one collection of values is a subtype of the most specific supertype of the other collection. For example, the plan *large-plan-b* uses only assault forces of type SOF-UNIT while the plan *large-plan-c* uses a more general set of forces (i.e., of type FORCE-COMPOSITION); in contrast, the plan *large-plan-b* uses a range of watercraft to fill the role WATER-ASSET while the plan *large-plan-c* uses only values of type BOAT. *Restricted subset* weakens the *restricted subtype* condition to require only that one collection of values be a subset of the other. Finally, *unstructured overlap* covers the case where there is no interesting subtype or subset relationship among the two collections of fill values.

Role differencing can provide interesting insights into fundamental differences between plans, as illustrated in Section 3.6.3. However, there are some limitations on its usefulness, as described further below.

The significance of role differences may be difficult to gauge in isolation. So, as noted above, the decision to use force Green-ODA-1 rather than Orange-ODA-2 to fill the ASSAULT-FORCE role is significant, as those two units have markedly different capabilities. However, the difference between the forces Green-ODA-1 and Green-ODA-2 is

insignificant as they have the same fundamental capabilities.⁸ This problem could be addressed by introducing some notion of 'semantic distance' between individuals.

Experimentation with our test suite showed that the difference types listed in Figure 10 have varying levels of significance. *Different single-valued, disjoint types* and *restricted subtype* seemed to be the most discriminating, in terms of identifying significant and coherent differences between plans.

The utility of role differencing can decrease as plans grow in size. The diminished utility derives from the fact that increased plan size can lead to increased numbers of occurrences of a role that are not closely related. So, for example, it is possible to introduce multiple assault forces that are inserted at different drop locations for larger-scale plans. Thus, while unrestricted role differencing can be useful in small- to medium-sized plans, larger plans would benefit from some scheme to contextualize role fills to certain portions of the plan.

3.6.3 Sample Plan Comparison

Figure 11 displays the results of applying our metatheoretic plan comparison techniques to the two test plans *medium-plan-a* and *medium-plan-b*.

In looking at the results of task feature differencing, two fundamental differences emerge: *medium-plan-a* contains refueling activities and *medium-plan-b* does not, while *medium-plan-b* contains SEAD (suppression of enemy air defense) activities and *medium-plan-a* does not. In terms of template feature differencing, there is some variation among expected values for key evaluation criteria. Given the use of a fairly coarse-grained set of qualitative values for template feature domains in the SOF metatheory, the scope for variability is limited. A more fine-grained set of values would enable more precise comparisons.

Task Feature Differencing:

Task Features in *medium-plan-a* but not in *medium-plan-b*: REFUELING

Task Features in *medium-plan-b* but not in *medium-plan-a*: SEAD

Template Feature Differencing:

Template Feature	Value for <i>medium-plan-a</i>	Value for <i>medium-plan-b</i>
DURABILITY	medium	low
FORCE-FATIGUE	high	medium
FORCE-INTEGRITY	high	medium
LZ-PREP-DEMAND	medium	low

⁸ Other factors, such as where the forces are based, could also impact their significance. Here, we assume that different units of the same type are located at the same base.

Role Differencing:

Different Single-valued

Role	Value for <i>medium-plan-a</i>	Value for <i>medium-plan-b</i>
HELICOPTER	uh-601-2	mh-60-g-pave-hawk-2
RECON-ASSET	satellite-1	green-oda-2

Disjoint Multi-valued

Role	Values for <i>medium-plan-a</i>	Values for <i>medium-plan-b</i>
ASSET	csar-c2-b tanker-1	uh-601-1 yorktown sead-1 csar-cl-a
EXFIL-ASSET	uh-601-1	aa201 uh-60a-2
FIRE-SUPPORT-ASSET	av-8b-harrier-ii-a	yorktown ah-100-1 ch-53e-super-stallion-1
INFIL-ASSET	uh-601-2	aa7864 mh-60-g-pave-hawk-2
TRANSPORT-ASSET	tanker-1 av-8b-harrier-ii-a uh-601-1 uh-601-2	sead-1 uh-60a-2 mh-60-g-pave-hawk-2

Restricted Subtype

Role	Type for <i>medium-plan-a</i>	Type for <i>medium-plan-b</i>
ASSAULT-FORCE	ORANGE-UNIT	SOF-UNIT
FORCE	ORANGE-UNIT	SOF-UNIT
INFIL-POINT	BUILDING	POINT-LOCATION
INFILTRATION-TEAM	ORANGE-UNIT	SOF-UNIT
LANDING-LOCATION	AIRPORT	POINT-LOCATION

Subset

Role	Values for <i>medium-plan-a</i>	Values for <i>medium-plan-b</i>
EXFIL-POINT	mogadishu-town-hall	mogadishu-town-hall mogadishu-building4
INFILTRATION-START	riyadh-airport	riyadh-airport ankara-airport

Figure 11. Metatheoretic Comparison of Plans *medium-plan-a* and *medium-plan-b*

Role differencing highlights some interesting variations in the use of resources among the two plans. (For simplicity, we have omitted the *unstructured overlap* category, as it is the least interesting of the possible role differences and generally the most common.) Both plans include reconnaissance operations, but *medium-plan-a* relies on a satellite (satellite-1) while *medium-plan-b* makes use of a ground force (green-oda-2) as the asset used to perform the reconnaissance. This difference can be quite significant, as the type of reconnaissance asset impacts the nature and quality of intelligence information that can be obtained. Different types of helicopters and transport assets are used, each with their individual strengths and weaknesses. In terms of overall asset usage, *medium-plan-a* uses a much narrower set of values than does *medium-plan-b* as evidenced by the fact that

the fills for the roles `ASSET`, `EXFIL-ASSET`, `FIRE-SUPPORT-ASSET`, and `INFIL-ASSET` in *medium-plan-a* are a restricted subset of those in *medium-plan-b*.

Overall, a user looking at the style of comparison in Figure 11 could quickly grasp the fundamental differences in strategic and resource usage between the two plans. Detailed examination of the plans themselves shows that there are additional differences in terms of unimportant low-level activities used to accomplish higher-level tasks and resource allocation. However, the metatheoretic comparison hides these nonessential differences.

3.7 Plan Space Analysis

In addition to supporting reasoning about individual and pairs of plans, the domain metatheory can also be used as the basis for reasoning about collections of plans. Our work explored two capabilities of this type: identifying unique characteristics of a plan, and identifying maximally different plans.

3.7.1 Identifying Unique Characteristics of a Plan

When considering a given plan relative to a set of candidate solutions, we can use the metatheoretic differencing capabilities to identify three distinguishing characteristics of a plan *P* relative to a set *S* of candidates.

1. **Unique task features:**
 - *P* has a task feature not found in any other solution in *S*.
 - *P* lacks a task feature found in all other solutions in *S*.
2. **Unique normalized template features:** *P* has a normalized template feature value that differs from the corresponding template feature value for all other solutions in *S*. This case is especially interesting when all other plans share a common value for that template feature.
3. **Differing role fills:** There is a role common to all plans for which some fill value in *P* does not occur as a fill value in other solutions in *S*.

In our plan comparison toolkit, we have implemented capabilities to support (1) and (2), but not (3).

Figure 12 summarizes the unique task features and normalized template features found for our suite of test plans, which occurred in the plans *small-plan-b* and *medium-plan-a*.

The plan *small-plan-b* differs from all other plans in the test suite on the value for the template feature `BLUE-CASUALTY-RISK`. In particular, its value for that feature is `low` while all other plans in the test suite have value `medium`.

Plan: *small-plan-b*

Has Unique Template Feature Values:

`BLUE-CASUALTY-RISK`: `low` (all others have value `medium`)

Plan: *medium-plan-a*

Has Unique Task features: REFUELING

Has Unique Template Feature Values:

DURABILITY: medium

FORCE-FATIGUE: high

LANDING-ZONE-PREPARATION-DEMAND: medium (all others have value medium)

Figure 12. Unique Task and Template Features from the Test Suite

The plan *medium-plan-a* has several unique characteristics relative to the other plans in the test suite. First, it is the only plan with the task feature REFUELING; hence, no other plans in the test suite include refueling operations. Second, while the plan *medium-plan-a* has the value medium for the template feature LANDING-ZONE-PREPARATION-DEMAND, all other plans have the value low. Finally, the plan *medium-plan-a* differs from all the other plans in the values for the template features DURABILITY and FORCE-FATIGUE; in those cases, however, there is no common value for the remaining plans in the test suite.

3.7.2 Maximally Different Plans

For many applications, a human planner will want to explore a range of plans that embody qualitatively different solutions to the problem at hand [Tate et al. 1998; Myers & Lee 1999]. Such exploration can be useful both in terms of helping the user understand fundamental tradeoffs that must be made, and identifying ‘out of the box’ solutions that he may not normally consider.

Our metatheoretic differencing techniques can be used as the basis for identifying plans that are semantically ‘far apart’ from each other, and hence are likely to have significant qualitative differences. To that end, we define a concept of *distance* between plans that builds on the concepts of *task feature*, *template feature*, and *role distance* between plans.

3.7.2.1 Task Feature Plan Distance

Our definition of task feature distance constitutes a normalized form of Hamming distance for the task features within the plans. In particular, task feature distance is defined to be the ratio of the number of task features that appear in one but not both plans to the number of features that appear in either plan.

Definition 3 [Task Feature Plan Distance] The *task feature distance* between plans P_1 and P_2 , denoted by $TaskFeatureDistance(P_1, P_2)$, is defined by the formula

$$TaskFeatureDistance(P_1, P_2) = \frac{|TaskFeatures(P_1) - TaskFeatures(P_2)| + |TaskFeatures(P_2) - TaskFeatures(P_1)|}{|TaskFeatures(P_1) \cup TaskFeatures(P_2)|}$$

3.7.2.2 Template Feature Plan Distance

Template feature distance for a pair of plans is defined to be the average difference between the values for those features that are common to both plans, normalized with

respect to the range of possible values for the features. Given that the template feature values are defined in qualitative terms, we perform the differencing and normalizations with respect to the mapping θ_f^{-1} of the qualitative values to the designated real interval for that feature, $Interval(f)$. Let $TemplateFeatures(P)$ denote the set of template features that occur in plan P .

Definition 4 [Template Feature Plan Distance] The *task feature distance* between plans P_1 and P_2 , denoted by $TemplateFeatureDistance(P_1, P_2)$, is defined by the formulae

$$TemplateFeatureDistance(P_1, P_2) = \frac{\sum_{f \in F^\cap} \frac{TemplateFeatureDiff(f, P_1, P_2)}{Interval(f)}}{|F^\cap|}$$

$$F^\cap = TemplateFeatures(P_1) \cap TemplateFeatures(P_2)$$

$$TemplateFeatureDiff(f, P_1, P_2) = \frac{(|I_2| \times \sum_{v \in I_1} \theta_f^{-1}(v)) - (|I_1| \times \sum_{v \in I_2} \theta_f^{-1}(v))}{|I_1| \times |I_2|}$$

$$I_1 = TemplateFeatureInsts(f, P_1)$$

$$I_2 = TemplateFeatureInsts(f, P_2)$$

3.7.2.3 Role Plan Distance

Role distance for a pair of plans is defined in terms of how distant the sets of fill values are for the roles that the two plans have in common. Our measure for the distance between sets of role fill values consists of the ratio of values that appear in one but not both sets of fills to the total number of fill values (another normalized form of Hamming distance). We note that when possible it may be appropriate to employ more specialized definitions that take into account the semantics of the underlying values. Such a definition could, for instance, reflect the fact that two airplanes of the same type are ‘closer’ than an airplane and a helicopter.

Definition 4 [Role Plan Distance] The *role distance* between plans P_1 and P_2 , denoted by $TemplateFeatureDistance(P_1, P_2)$, is defined by the following formulae, where $Roles(P)$ denotes the set of roles that occur in plan P .

$$RoleDistance(P_1, P_2) = \frac{\sum_{r \in R^\cap} RoleFillDifference(r, RoleFills(r, P_1), RoleFills(r, P_2))}{|R^\cap|}$$

$$RoleFillDifference(r, V1, V2) = \frac{|V1 - V2| + |V2 - V1|}{|V1 \cup V2|}$$

$$R^\cap = Roles(P_1) \cap Roles(P_2)$$

3.7.2.4 Metatheoretic Plan Distance

Using the above definitions, we can define the *metatheoretic distance* between two plans.

Definition 2 [Metatheoretic Plan Distance] The *metatheoretic distance* between plans P_1 and P_2 , denoted by $PlanDistance(P_1, P_2)$, is defined as

$$PlanDistance(P_1, P_2) = Weight_{TaskFeatures} _ TaskFeatureDistance(P_1, P_2) \\ + Weight_{TemplateFeatures} _ TemplateFeatureDistance(P_1, P_2) \\ + Weight_{Roles} _ RoleDistance(P_1, P_2)$$

where $Weight_{TaskFeatures} + Weight_{TemplateFeatures} + Weight_{Roles} = 1$.

The definition of metatheoretic plan distance assumes a set of weights that can be used to adjust the relative importance of task features, template features, and roles in the distance specification. Because these three components address different aspects of an overall plan, different users may be interested in biasing the plan distance calculation to stress the relative importance of these three components.

Similarly, the definitions for template feature, task feature, and role distance can be modified in a straightforward manner to support the incorporation of weights that enable different features and roles to be given different degrees of emphasis. Such weights could be defined either for an entire domain or customized by an individual user (on a situation-by-situation basis, if so desired).

Plan Distance	Template Feature Distance	Task Feature Distance	Role Distance	Plan1	Plan2
.08	.03	.11	.08	large-plan-a	large-plan-b
.09	.00	.00	.28	large-plan-a	large-plan-c
.12	.00	.00	.35	tiny-plan-a	tiny-plan-b
.15	.03	.11	.30	large-plan-b	large-plan-c
.15	.15	.00	.31	small-plan-a	small-plan-b
.17	.12	.14	.26	small-plan-a	medium-plan-a
.21	.00	.14	.49	small-plan-a	medium-plan-b
.26	.27	.14	.37	small-plan-b	medium-plan-a
.29	.15	.14	.56	small-plan-b	medium-plan-b
.31	.14	.25	.53	medium-plan-a	medium-plan-b
.34	.12	.25	.67	small-plan-a	large-plan-a
.35	.07	.22	.75	medium-plan-b	large-plan-b

.35	.21	.67	.18	<i>tiny-plan-b</i>	<i>small-plan-b</i>
.36	.12	.25	.72	<i>small-plan-a</i>	<i>large-plan-c</i>
.37	.08	.33	.70	<i>small-plan-a</i>	<i>large-plan-b</i>
.40	.27	.25	.67	<i>small-plan-b</i>	<i>large-plan-c</i>
.40	.18	.33	.69	<i>medium-plan-a</i>	<i>large-plan-a</i>
.40	.27	.25	.68	<i>small-plan-b</i>	<i>large-plan-a</i>
.41	.18	.33	.71	<i>medium-plan-a</i>	<i>large-plan-c</i>
.41	.14	.40	.69	<i>medium-plan-a</i>	<i>large-plan-b</i>
.41	.11	.33	.79	<i>medium-plan-b</i>	<i>large-plan-a</i>
.42	.23	.33	.69	<i>small-plan-b</i>	<i>large-plan-b</i>
.42	.11	.33	.81	<i>medium-plan-b</i>	<i>large-plan-c</i>
.44	.21	.67	.45	<i>tiny-plan-a</i>	<i>small-plan-b</i>
.47	.29	.67	.46	<i>tiny-plan-a</i>	<i>small-plan-a</i>
.47	.29	.67	.46	<i>tiny-plan-b</i>	<i>small-plan-a</i>
.51	.29	.71	.51	<i>tiny-plan-b</i>	<i>medium-plan-b</i>
.51	.29	.71	.53	<i>tiny-plan-a</i>	<i>medium-plan-b</i>
.52	.42	.71	.42	<i>tiny-plan-a</i>	<i>medium-plan-a</i>
.52	.42	.71	.42	<i>tiny-plan-b</i>	<i>medium-plan-a</i>
.56	.33	.75	.58	<i>tiny-plan-a</i>	<i>large-plan-c</i>
.56	.33	.75	.60	<i>tiny-plan-a</i>	<i>large-plan-a</i>
.56	.33	.75	.60	<i>tiny-plan-b</i>	<i>large-plan-a</i>
.57	.33	.75	.62	<i>tiny-plan-b</i>	<i>large-plan-c</i>
.58	.37	.78	.60	<i>tiny-plan-a</i>	<i>large-plan-b</i>
.58	.37	.78	.60	<i>tiny-plan-b</i>	<i>large-plan-b</i>

Figure 13. Metatheoretic Plan Distances for the SOF Test Suite

3.7.2.5 Application of Metatheoretic Plan Distance to the Test Suite

The motivation for introducing the concept of plan distance was to support a user in identifying plans that are semantically distinct from each other. The results in Figure 13 show that, at least for the SOF domain, our definition of metatheoretic plan distance is effective. The figure displays the distance calculations for our test suite of plans, using an equal distribution of weights for the task feature, template feature, and role distance calculations (i.e., $Weight_{TaskFeatures} = Weight_{TemplateFeatures} = Weight_{Roles} = 1/3$).

As one might expect, the distance calculations show that the ‘closest’ plans correspond to core plans and their variants. In particular, the shortest plan distances found are between the two tiny plans (.08), between the various large plans (.08, .09, .15), and between the two small plans (.15). The distance between the two medium plans is appreciably higher (.31); as noted in Figure 4 and made apparent in Figure 11, these two plans are not simple variants of each other but rather contain key strategic differences. In addition, the plans that are farthest apart (the tiny vs. large plans) are indeed the plans with the greatest meaningful variations among them.

3.8 Discussion

Efforts to date on developing general-purpose plan summarization and comparison tools have focused on approaches that analyze plan structures and planning decisions directly. Such approaches suffer from the problem that those syntactic elements do not necessarily shed light on the semantic content of a plan. In particular, it is possible to have plans with significant variations in syntactic structure that are semantically similar; as well, plans with similar syntactic structure may have semantic differences that are extremely significant.

One of the key benefits of our qualitative approach to plan summarization and comparison is its emphasis on semantic rather than syntactic characteristics of plans. Thus, our comparison of metatheoretic properties grounds the results in concepts that are intended to be both significant and of interest to the user, rather than concepts that are important to an automated system when generating a plan.

Our qualitative approach to plan comparison and summarization is not intended to eliminate the need for more precise, quantitative analysis tools. Rather, we envision the qualitative approach being valuable to a user in the early stages of planning, in terms of enabling the user both to quickly understand the main features of a plan, and to perform an inexpensive analysis of what differentiates alternative candidate plans. After developing some preliminary understanding of individual plans and their relative strengths and weaknesses, a user may then wish to perform more expensive and time-consuming quantitative analyses to assess plans in detail.

Another benefit of our approach is that it readily supports customization to domains, individual users, or specific contexts. This can be achieved by defining filters on the sets of features and roles that are of interest to the user (for plan summarization and plan comparison) and by appropriate adjustment of weights (for analyzing a solution space).

We believe that the experimental analysis of our plan summarization and comparison methods within the SOF domain shows that the techniques are effective for helping a user understand subtle aspects of individual plans, importance differences among plans, and the structure of the overall solution space. Our efforts have, however, identified some areas for future work.

- The existence of a well-designed domain metatheory is critical for the successful application of our plan summarization and comparison methods. As noted elsewhere [Myers 2000], the design of the metatheory should be a by-product of a principled approach to modeling a planning domain. Still, it remains a bit of an art to design a metatheory appropriately.
- The framework presented here assumes an HTN paradigm for the underlying plan development. Conceptually, it would be straightforward to map the framework to a classical operator-style planning paradigm. It would be interesting to consider specializations of the framework for other types of planners (e.g., temporal planners).

- As noted above, the explanatory capability of our methods when applied to larger plans could be improved by introducing some capability for contextualization. Such a capability would enable generalizations to be localized to meaningful subportions of a plan rather than spanning the entire plan. This localization could enable more interesting regularities or trends within plans to be identified. The hierarchical structure of HTN plans provides an obvious way to generate candidate contexts, namely, subplans appearing below a given task node. Within that framework, however, identifying the most appropriate contexts to use in a given situation remains an interesting challenge.

4 Conclusions

For mixed-initiative planning tools to gain acceptance by users in real applications, they must provide effective explanation capabilities. To date, however, there has been relatively little effort devoted to developing such capabilities; furthermore, the work that has been done has suffered from two shortcomings. First, it has focused on syntactic elements of plans despite the fact that such syntactic structures may not correspond to important semantic features. Second, the methods have by and large ignored the importance of tailoring explanations toward a user. One manifestation of this second problem is the tendency toward explaining a plan in terms of the system's derivation trace, rather than a user's perspective of what is or is not important.

On this project, we developed plan explanation capabilities that both are grounded in semantic aspects of plans and explicitly take into account the perspective of the user who is requesting the explanation. We focused on explanatory capabilities along two complementary dimensions: explaining schedules using examples and plan summarization and comparison.

Our approach to explaining schedules with examples contrasts with traditional approaches to explanation that present the chain of reasoning the system used to produce the solution. In the example-based approach, the explainer justifies the system's reasoning by generating examples of alternative solutions, selecting from these examples to create a diverse set for presentation, and justifying the system's solutions with respect to each of the examples. Our implementation of the approach generates examples using sample output of SRI's Calendar Manager.

In the area of plan summarization and comparison, we developed a qualitative approach that builds on the notion of a domain metatheory for a given planning domain. This approach has the benefit of framing summaries and comparisons in terms of high-level semantic concepts, rather than low-level syntactic details. We developed a set of tools that instantiate this approach and evaluated them within the context of a rich special operations planning domain. That evaluation showed that the approach is indeed useful for helping a user to understand both individual plans and tradeoffs among plans.

Overall, much work remains to be done to develop a better understanding of how to build effective explanation tools for mixed-initiative planning systems. We believe, however, that our work on this project represents an important first step toward achieving that long-term objective.

5 Acknowledgments

The authors thank Mabry Tyson for his assistance in making modifications to PASSAT to support the metatheoretic summarization and comparison capabilities. They also thank Peter Jarvis for developing the suite of test plans used to evaluate the plan summarization and comparison techniques.

6 References

- Ai-Chang, M., Bresina, J., Charest, L., Jonsson, A., Hsu, J., Kanefsky, B., Maldague, P., Morris, P., Rajan, K., and Yglesias, J. MAPGEN: Mixed Initiative Planning and Scheduling for the Mars '03 MER Mission. In *Proceedings of the AAAI Spring Symposium on Human Interaction with Autonomous Systems in Complex Environments*, AAAI Technical Report SS-03-04, 2003.
- Berry, P. M., Gervasio, M., Uribe, T. E., Myers, K. L., and Nitz, K. A Personalized Calendar Assistant. In *Proceedings of the AAAI Spring Symposium on Human Interaction with Autonomous Systems in Complex Environments*, AAAI Technical Report SS-03-04, 2003.
- Buchanan, B. G. and Shortliffe, E. H. (Eds.). *Rule-Based Expert Systems: The MYCIN Experiments of the Stanford Heuristic Programming Project*. Addison-Wesley Publishing Company, Reading, MA, 1984.
- Clancey, W. J. The Epistemology of a Rule-Based Expert System -- A Framework for Explanation, *Artificial Intelligence*, Vol. 20, 1983.
- Erol, K., Hendler, J., and Nau, D. (1994). Semantics for Hierarchical Task-Network Planning. Technical Report CS-TR-3239, Computer Science Department, University of Maryland.
- Frank, J. and Jónsson, A. Constraint Based Attribute and Interval Planning. *Journal of Constraints*, 8(4), 339-364, 2003.
- McGuinness, D. and da Silva, P. Infrastructure for Web Explanations. In *Proceedings of 2nd International Semantic Web Conference (ISWC2003)*, Sanibel Is., FL, Springer, October 2003.
- Myers, K. Strategic Advice for Hierarchical Planners. In *Proceedings of the 5th International Conference on Principles of Knowledge Representation and Reasoning*, Cambridge, MA, 1996.
- Myers, K. L. and Lee, T. J. Generating Qualitatively Different Plans through Metatheoretic Biases. In *Proceedings of the Sixteenth National Conference on Artificial Intelligence (AAAI-99)*, AAAI Press, Menlo Park, CA, 1999.
- Myers, K. L., Tyson, W. M., Wolverton, M. J., Jarvis, P. A., Lee, T. J., and desJardins, M. PASSAT: A User-centric Planning Framework. In *Proceedings of the 3rd International NASA Workshop on Planning and Scheduling for Space*, Houston, TX, 2002.
- Myers, K. L., Jarvis, P., Tyson, W. M., and Wolverton, M. J. A Mixed-initiative Framework for Robust Plan Sketching. In *Proceedings of the 13th International Conference on Automated Planning and Scheduling*, Trento, Italy, 2003.
- Sqalli, M. and Freuder, E. Inference-Based Constraint Satisfaction Supports Explanation in *Proceedings of the 13th National Conference on Artificial Intelligence (AAAI 96)*, 1996.
- Swartout, W., Paris, C., and Moore, J. Design for Explainable Expert Systems, *IEEE Expert*, Vol. 6, Number 3, 58-64, 1991.
- Tate, A., Dalton, J. and Levine, J. Generation of Multiple Qualitatively Different Plan Options. In *Proceedings of Artificial Intelligence Planning Systems*, pages 27-35, 1998.
- Wick, M. R. and Thompson, W. B. Expert System Explanation, *Artificial Intelligence*, 54(1-2), 1992.
- Wilkins, D. E. Using the SIPE-2 Planning System, Technical Report, Artificial Intelligence Center, SRI International, Menlo Park, CA, 1993.
- Wolverton, M. J. Presenting Significant Information in Expert System Explanation. In *Proceedings of the Seventh Portuguese Conference on Artificial Intelligence*, 1995.

Young, R. M. Cooperative Plan Identification: Constructing Concise and Effective Plan Descriptions. In *Proceedings of the National Conference of the American Association for Artificial Intelligence*, Orlando, FL, August 1999.